

Unit 2, Topic 7 – Understanding the Functionality of HTM

European Institute of Innovation, Entrepreneurship and Technology
Lessingstraße 79, 13158 Berlin, Germany

Presenter Name| Irikefe Hope Oniovosa



**Lessingstraße 79,
13158 Berlin, Germany
Phone: +49 30 233618555
Fax: +49 30 233618556
Email: info@eiiet.com**

Discover the world of eiIET

1.0 Introduction to HTML Structure

- **HTML as a Foundation:**

HTML (HyperText Markup Language) is the fundamental language used to create and structure content on the web. Every website starts with HTML, which tells the browser how to display text, images, links, and other media.

- **Role for Designers and Developers:**

Web designers and developers rely on HTML to lay out the structure of a webpage. It's the skeleton on which all styling (via CSS) and behavior (via JavaScript) are built.

Key Terms in HTML

1. Doctypes:

1. **Definition:**

The doctype is a declaration at the very beginning of an HTML document. It tells the browser which version of HTML is being used.

Example: `<!DOCTYPE html>`

Elements and Tags:

- **Elements:**

The building blocks of an HTML document. Each element represents a part of the document (such as a paragraph, heading, or image).

- **Tags:**

HTML elements are marked up with tags. Tags usually come in pairs: an opening tag (e.g., <p>) and a closing tag (e.g., </p>), which enclose the content.

- **Example:** <h1>This is a Heading</h1>

<p>This is a paragraph of text.</p>

- **Attributes:**

- **Definition:**

Attributes provide additional information about an element. They are placed within the opening tag.

- **Examples:**

- class and id are used for styling and scripting.
- src for images or href for links.

- **Example:**

Declaring Document Language

- **Declaring Document Language**

- **Purpose:**

Declaring the language of the document helps search engines and browsers understand the primary language of your content.

- **How to Declare:**

Include the lang attribute in the <html> tag.

- **Example:** `<html lang="en">`

The Head Element

- **Purpose of <head>:**

The <head> element contains metadata and information that isn't directly displayed on the page. It includes:

- **Metadata:** Information like character encoding (e.g., <meta charset="UTF-8">).
- **Document Title:** Specified with the <title> tag.
- **Scripts and Styles:** Links to external files (like CSS and JavaScript) and inline code.
- **Other Elements:** Links to favicons, meta tags for SEO, etc.

- **Example:**

```
<head>  
  <meta charset="UTF-8">  
  <title>My Web Page</title>  
  <link rel="stylesheet" href="styles.css">  
  <script src="script.js"></script>  
</head>
```

Organizing HTML Syntax: Best Practices

1. Lowercase Letters:

1. Always use lowercase for element names, attribute names, and values.
2. **Example:** `<div class="container">` instead of `<DIV CLASS="container">`.

2. Indentation:

1. Indent nested elements to improve readability and maintain a clear structure.
2. **Example:**

```
<ul>
  <li>Item 1</li>
  <li>Item 2</li>
</ul>
```

Organizing HTML Syntax: Best Practices

- **Double Quotes for Attribute Values:**

- Always enclose attribute values in double quotes.
- **Example:** `<a href="https://example.com">Visit Example</a>`

- **Boolean Attributes:**

- For Boolean attributes (attributes that are either present or not), you can omit the value.
- **Example:** `<input type="checkbox" checked>` is acceptable.

- **Self-Closing Elements:**

- Avoid unnecessary forward slashes in self-closing elements in HTML5.
- **Example:** `` is preferred over ``.

Global Attributes

- **Definition:**

Global attributes are attributes that can be applied to any HTML element. They provide additional functionality or styling.

- **Examples and Their Uses:**

- **class:** Used to assign one or more class names to an element for CSS styling.
- **id:** A unique identifier for an element, useful for targeting specific elements with CSS or JavaScript.
- **style:** Inline CSS styles for an element.
- **accesskey:** Provides a shortcut key to focus or activate an element.
- **lang:** Specifies the language of an element's content.
- **tabindex:** Controls the order of focus when tabbing through elements.
- **data-*:** Custom data attributes that can be used to store extra information.
- **hidden:** Hides the element from the user.

- **Example:**

```
<div id="main" class="container" style="background-color: #fofofo;" data-role="page">  
  <!-- Content here -->  
</div>
```

Summary

- **HTML Structure:** Forms the foundation of web pages by structuring content using elements and tags.
- **Key Components:** Include doctypes, elements, tags, attributes, and the <head> element.
- **Syntax Best Practices:** Use lowercase, proper indentation, double quotes for attribute values, and streamlined self-closing elements.
- **Global Attributes:** Enhance elements with additional functionality and are applicable across the entire HTML document.

This overview provides a comprehensive understanding of HTML document structure, setting the stage for further lessons on web development. Feel free to ask questions or request further examples as we continue our exploration of HTML!

Structural Markup

A. Block-Level vs. Inline Elements

1. Block-Level Elements:

1. Definition:

Block-level elements start on a new line and take up the full available width by default. They are used to structure larger sections of content.

2. Examples:

1. `<div>`: A generic container for flow content.
2. `<p>`: Defines a paragraph.
3. `<h1>` to `<h6>`: Define headings.
4. `<article>`, `<section>`, `<header>`, `<footer>`: Used for semantic structure.

3. When to Use:

Use block-level elements when you need to group content into distinct sections or when you want content to appear on its own line.

A. Block-Level vs. Inline Elements_2

2. Inline Elements:

1. Definition:

Inline elements do not start on a new line; they only take up as much width as necessary. They are used for small pieces of content within block-level elements.

2. Examples:

1. `<span>`: A generic inline container.
2. `<a>`: Defines a hyperlink.
3. `<strong>` and `<em>`: Used to emphasize text.

3. When to Use:

Use inline elements for styling or marking up parts of a text, such as words or phrases, without disrupting the flow of the content.

B. Content Models

- **B. Content Models**

- **Content Models:**

HTML elements can be categorized into different content models based on the type of content they can contain or how they interact with other elements. Some common models include:

- **Flow Content:**

General content that can include both block-level and inline elements (e.g., `<div>`, `<p>`, `<span>`).

- **Sectioning Content:**

Elements that define the structure of a document (e.g., `<section>`, `<article>`, `<nav>`).

- **Heading Content:**

Elements used for headings (e.g., `<h1>` to `<h6>`).

- **Phrasing Content:**

Elements that define the text and its inline semantics (e.g., `<span>`, `<a>`, `<em>`, `<strong>`).

- **Embedded Content:**

Elements that embed external content (e.g., `<img>`, `<video>`, `<iframe>`).

- **Interactive Content:**

Elements that respond to user interactions (e.g., `<button>`, `<input>`, `<a>` with href).

C. Structuring and Formatting Text_1

1. Articles, Sections, Headings, and Paragraphs:

1. Usage:

1. **<article>**: Represents a self-contained composition (e.g., a blog post or news article).
2. **<section>**: Groups related content together.
3. **<h1> to <h6>**: Define the hierarchy of headings; <h1> is the most important.
4. **<p>**: Defines a paragraph of text.

```
<article>
<header>
  <h1>Introduction to Web Development</h1>
</header>
<section>
  <h2>HTML Basics</h2>
  <p>HTML stands for HyperText Markup Language and is the backbone of web content.</p>
</section>
<footer>
  <p>Published on April 3, 2025</p>
</footer>
</article>
```

C. Structuring and Formatting Text_2

- **Thematic Breaks, Emphasis, and Importance:**

- **Thematic Breaks:**

Use `<hr>` to insert a horizontal rule, which indicates a thematic change or a break in content.

- **Emphasis and Importance:**

- `<em>`: Emphasizes text (usually rendered in italics).
- `<strong>`: Indicates strong importance (usually rendered in bold).

```
<p>This is <em>emphasized</em> text and this is <strong>very important</strong> text.</p>
<hr>
```

D. Creating Lists_1

1.Unordered Lists:

1. **Tags:** for the list, for list items.

```
<ul>  
<li>Apples</li>  
<li>Oranges</li>  
<li>Bananas</li>  
</ul>
```

• Ordered Lists:

- **Tags:** for the list, for list items.
- **Attributes:** Use start to specify the starting number and type to change the numbering style.
- **Example:**

```
<ol start="1" type="1">  
<li>First step</li>  
<li>Second step</li>  
<li>Third step</li>  
</ol>
```

D. Creating Lists_2

- **Definition Lists:**

- **Tags:** <dl> for the list, <dt> for the term (definition term), and <dd> for the description (definition description).

```
<dl>
  <dt>HTML</dt>
  <dd>HyperText Markup Language.</dd>
  <dt>CSS</dt>
  <dd>Cascading Style Sheets.</dd>
</dl>
```

- **Nested Lists:**

- **Usage:** You can nest lists inside one another by placing an or inside an .

```
<ul>
  <li>Fruits
    <ul>
      <li>Apples</li>
      <li>Oranges</li>
    </ul>
  </li>
  <li>Vegetables</li>
</ul>
```

E. Creating Links_1

- **Internal Links:**

- **Definition:** Links that navigate to another section within the same page.

- Example: `<a href="#section1">Go to Section 1</a>`

- **Anchor Setup:** Use the id attribute on the target element:

- `<h2 id="section1">Section 1</h2>`

- **External Links:**

- **Definition:** Links that navigate to a different website.

- `<a href="https://www.example.com">Visit Example</a>`

- **Email Links:**

- **Definition:** Use the mailto: protocol to create a link that opens the user's email client.

- **Example:** `<a href="mailto:someone@example.com">Email Us</a>`

E. Creating Links_2

- **Email Links:**

- **Definition:** Use the mailto: protocol to create a link that opens the user's email client.

- **Example:** `<a href="mailto:someone@example.com">Email Us</a>`

- **Links that Open in a New Window/Tab:**

- **Attribute:** Use target="_blank" to open links in a new window or tab.

- **Example:** `<a href="https://www.example.com" target="_blank">Visit Example in a New Tab</a>`

- **Linking to Specific Parts:**

- **Within the Same Page:** Use anchor tags with href linking to an id on another element.

- **Example:**

```
<a href="#contact">Contact Us</a>
```

```
<section id="contact">
```

```
  <h2>Contact Information</h2>
```

```
  <p>Our address and phone number.</p>
```

```
</section>
```

Summary

- **Block-Level vs. Inline Elements:**

Use block-level elements for large sections (e.g., <div>, <p>) and inline elements for small pieces of content (e.g., , <a>).

- **Content Models:**

Understand the different content models (flow, sectioning, heading, phrasing, embedded, interactive) to choose appropriate elements.

- **Text Structuring:**

Organize content with articles, sections, headings, paragraphs, thematic breaks, and emphasis tags.

- **Lists:**

Create unordered, ordered, definition, and nested lists to present content in an organized manner.

- **Links:**

Learn to create internal, external, email links, and links that open in a new tab; also link to specific parts within a page using anchor tags.

Page Components - A. Adding Images

1. Using the Tag:

1. **Purpose:** The tag embeds images into an HTML page.
2. **Key Attributes:**
 - **src:** Specifies the image file location.
 - **alt:** Provides alternative text for accessibility if the image cannot be displayed.
 - **Example:**

2. Considerations for Images:

1. **File Formats:** Common formats include JPEG, PNG, GIF, and SVG.
2. **Image Size and Resolution:** Use appropriately sized images to balance quality and page load speed.
3. **Maintaining Original Proportions:** Use CSS or HTML attributes (like width and height) carefully to avoid distortion.

3. Positioning Images:

1. Images can be positioned using CSS (e.g., float, margin, padding).
2. The alt attribute is essential for accessibility, ensuring that screen readers can describe the image to users with visual impairments.

B. Creating Tables

- **Table Structure:**

- **Elements:**

- `<table>`: The container for the entire table.
- `<tr>`: Table row.
- `<th>`: Header cell for columns (typically bold and centered).
- `<td>`: Standard table cell.
- `<tbody>`: Groups the main body content.
- `<tfoot>`: (Optional) Groups footer content.

- **Combining Cells:**

- **colspan**: Merges multiple columns into a single cell.

- **rowspan**: Merges multiple rows into one cell.

- **Example:**

```
<td colspan="2">Combined cell for two columns</td>
```

```
<table>
  <thead>
    <tr>
      <th>Product</th>
      <th>Price</th>
    </tr>
  </thead>
  <tbody>
    <tr>
      <td>Laptop</td>
      <td>$800</td>
    </tr>
    <tr>
      <td>Tablet</td>
      <td>$300</td>
    </tr>
  </tbody>
  <tfoot>
    <tr>
      <td colspan="2">End of list</td>
    </tr>
  </tfoot>
</table>
```

C. Building Forms_1

1. Form Structure and Controls:

1. **<form>**: The container element for a form.
2. **Input Elements:**
 - **<input>**: Various types (text, password, radio, checkbox, etc.).
 - **<textarea>**: For multi-line text input.
 - **<select>** and **<option>**: Drop-down lists.
 - **Buttons:**
<button>, **<input type="submit">**, etc.

```
<form action="submit_form.php" method="post">
  <label for="name">Name:</label>
  <input type="text" id="name" name="name">

  <label for="email">Email:</label>
  <input type="email" id="email" name="email">

  <label for="message">Message:</label>
  <textarea id="message" name="message"></textarea>

  <button type="submit">Submit</button>
</form>
```

C. Building Forms_2

- **Grouping Elements:**

- Use <fieldset> and <legend> to group related form controls and provide a caption for the group.

- **Example:**

```
<fieldset>
  <legend>Personal Information</legend>
  <!-- Form elements here -->
</fieldset>
```

- **How Forms Work:**

- **Data Submission:** When a form is submitted, the data is sent from the browser to a server via the URL (GET) or request body (POST).
- **Basic Form Validation:** Use HTML attributes (like required, minlength, etc.) or JavaScript to ensure data meets certain criteria before submission.

D. Embedding Multimedia

1. Using <audio> and <video> Tags:

1. **Purpose:** Embed audio and video content directly into a web page.

```
<audio controls>  
  <source src="audio-file.mp3" type="audio/mpeg">  
  Your browser does not support the audio element.  
</audio>
```

```
<video width="320" height="240" controls>  
  <source src="video-file.mp4" type="video/mp4">  
  Your browser does not support the video tag.  
</video>
```

- **Specifying Multiple File Formats:**
- Include multiple <source> tags within <audio> or <video> to ensure compatibility across different browsers.
- **Customizing Controls and Attributes:**
- Use attributes such as autoplay, loop, muted, and poster (for videos) to enhance user experience.

E. Using Inline Frames (iframes)

- E. Using Inline Frames (iframes)

1. Purpose of <iframe>:

1. Embeds another HTML document within the current document.
2. Useful for displaying external content, such as videos, maps, or other web pages, without leaving your site..

```
<iframe src="https://www.example.com" width="600" height="400" title="Embedded Content"></iframe>
```

- **Considerations:**

- Set appropriate width and height.
- Use the title attribute to provide a description for accessibility.
- Be mindful of security and performance implications when embedding external content.

Conclusion

- **E. Using Inline Frames (iframes)**

- 1. Purpose of <iframe>:**

1. Embeds another HTML document within the current document.
2. Useful for displaying external content, such as videos, maps, or other web pages, without leaving your site..

```
<iframe src="https://www.example.com" width="600" height="400" title="Embedded Content"></iframe>
```

- **Considerations:**

- Set appropriate width and height.
- Use the title attribute to provide a description for accessibility.
- Be mindful of security and performance implications when embedding external content.

Additional Resources

- **Videos**

- **HTML Crash Course for Beginners**

This video provides a comprehensive introduction to HTML, covering basics from document structure to common elements.

Watch it here: [HTML Crash Course for Beginners \(Traversy Media\)](https://youtu.be/UB1O3ofR-EE) “https://youtu.be/UB1O3ofR-EE”

- **Understanding HTML Document Structure**

This video explains the foundational aspects of HTML document structure, including the use of tags, attributes, and best practices.

Watch it here: [Understanding HTML Document Structure](https://www.youtube.com/watch?v=pQN-pnXPaVg) “https://www.youtube.com/watch?v=pQN-pnXPaVg”

- **Reading Material**

- **Introductory Web Development Textbooks:**

Look for chapters on HTML5 in books such as *"HTML and CSS: Design and Build Websites"* by Jon Duckett, which offers visually engaging and beginner-friendly explanations.

- **Online Tutorials on HTML5:**

Websites like freeCodeCamp and Codecademy offer interactive lessons and comprehensive guides on HTML5 concepts and best practices.

Thank you



Entrepreneurship and Technology

Lessingstraße 79,

13158 Berlin, Germany

Phone: +49 176 55524449

Email: info@eiiet.com